

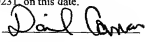
1954.65969

I hereby certify that this paper is being deposited with the United States Postal Service as Express Mail in an envelope addressed to: BOX PATENT APPLICATION, ASSISTANT COMMISSIONER FOR PATENTS, Washington, D.C. 20231 on this date.

January 23, 2002

Date

Express Mail No. EL846173622US



1 Inventors: Richard Marin; Joshua Landsman; Russ Bakke

2 METHOD AND SYSTEM FOR SECURING A COMPUTER
3 CONNECTED TO AN INSECURE NETWORK

4 [0001] The present invention generally relates to a method and
5 system for securing a computer connected to an insecure network when the
6 computer is not utilizing the insecure network. More specifically, it relates
7 to a method and system for securing a computer connected to an insecure
8 network when the computer is not utilizing the insecure network, wherein
9 the computer is installed with a program managing the connection with the
10 insecure network.

11 [0002] It is currently becoming more common for a typical
12 computer to be connected to multiple networks at any given time. For
13 example, a computer may be connected to an intranet via a local area
14 network (LAN) or/and the Internet via a Digital Subscriber Line (DSL), a
15 cable modem connection or a T connection. Because continuous

1 connection to the Internet (i.e., insecure network) using these various
2 connections is becoming the standard in the computer industry, a typical
3 computer is vulnerable to unwanted connections or intrusions, such as
4 hacker attacks, from the insecure network at any given time as long as the
5 computer is turned on and hooked up to the Internet. Thus, a method to
6 secure the computer from such unwarranted connections is needed to
7 protect the computer from any potentially damaging intrusions.

8 [0003] There are currently several commercially available
9 software programs, such as ZoneAlarm Pro® manufactured by ZoneLabs,
10 San Francisco, CA, McAfee Firewall® manufactured by Network
11 Associates®, Inc., Santa Clara, California, Norton Internet Security 2002®
12 manufactured by Symantec Corp.®, Cupertino, CA, Norton Personal
13 Firewall 2002® manufactured by Symantec Corp.®, Cupertino, CA and
14 BlackIce Defender® manufactured by Defender Network ICE
15 Corporation®, San Mateo, CA, that place a firewall between the computer
16 and the insecure network. In particular, the ZoneAlarm® program allows
17 users to decide which applications can and cannot use the Internet. An
18 Internet Lock is implemented in the ZoneAlarm® program for blocking
19 Internet traffic while the computer is unattended or while the Internet is not
20 being used. The McAfee firewall® program, on the other hand, filters all
21 the applications, system services, and protocols, including file and printer
22 shares (NetBIOS), IP protocols (TCP/IP, UDP/IP), service-based protocols
23 (FTP, Telnet), ARP/RARP, and Dynamic Host Configuration Protocol
24 (DHCP). Additionally, the firewall blocks the IPX and the NetBEUI on a
25 per device basis.

1 [0004] The Norton Internet Security® 2002 program and Norton
2 Personal Firewall® 2002 program offers a software program that blocks
3 incoming hack attacks while allowing trusted applications to connect to the
4 computer. Lastly, the BlackIce Defender® scans the DSL, cable modem or
5 dial-up Internet connection for hacker activity. When an attempted
6 intrusion is detected, the traffic from that source will be automatically
7 blocked. As a result, any unwanted intrusion is avoided. In all these
8 examples, the connection between the computer and the insecure network
9 remains connected. Basically, all of the prior solutions filter the connection
10 to the insecure network. In other words, while the computer is connected to
11 the insecure network, the known programs provide a security system in
12 front of the gateways or ports to the computer. The programs determine
13 whether a requesting source is trusted or untrusted, and only the trusted
14 sources are allowed access to the gateway or the ports.

15 [0005] The problem with these prior programs is that it is too
16 difficult to literally list or identify all the trusted sources. As a result, they
17 are generally riddled with multiple security leaks or shortcomings. As
18 shown, there is a need for an improved method for securing the computer
19 from the insecure network.

20 [0006] Accordingly, it is an object of the present invention to
21 provide an improved security program which more completely protects
22 computers from hazards borne by an insecure network.

23 BRIEF SUMMARY OF THE INVENTION

24 [0007] The present invention is directed to an improved method
25 and system for securing a computer connected to an insecure network when

1 the computer is not utilizing the insecure network. More specifically, it
2 relates to a method and system for securing a computer connected to an
3 insecure network when the computer is not utilizing the insecure network,
4 wherein the computer is installed with a program managing the connection
5 with the insecure network.

6 **[0008]** The present invention provides a method for securing a
7 computer connected to an insecure network when the computer is not
8 utilizing the insecure network, wherein the computer is installed with a
9 program managing the connection with the insecure network. The method
10 includes the steps of determining whether the computer is active,
11 deactivating the computer from the insecure network when it is determined
12 that the computer is inactive, and waiting for a predefined time period to
13 repeat the method.

14 **[0009]** Also, in another embodiment, the present invention
15 provides a computer program product comprising a computer readable code
16 stored on a computer readable medium that, when executed, the computer
17 program product causes a computer to determine whether the computer is
18 active, deactivate the computer from the insecure network when it is
19 determined that the computer is inactive, and wait for a predefined time
20 period to repeat the method.

21 DESCRIPTION OF THE DRAWINGS

22 **[0010]** FIG. 1 is a schematic diagram of a network system in
23 which the present method is implemented;

24 **[0011]** FIG. 2 is a flowchart illustrating an overall preferred
25 method of the present invention;

1 to the system would be greatly reduced by the present invention, and the
2 network security is improved.

3 **[0018]** A schematic diagram of a network system is shown in
4 FIG. 1, and indicated generally at 10. A computer 12 is shown to be
5 connected to the Internet 14 (i.e., insecure network) and a LAN 16 (secure
6 network) running an intranet via a computer server 18. As shown, there are
7 multiple computers 20, 22, 24, 26 including the computer 12, which are
8 referred to as client computers, connected to the computer server computer
9 18. The Internet 14 also shows multiple computers 28, 30, 32, 34, 36, 38,
10 40 including the computer 12. However, in practice, the Internet generally
11 includes millions of computers connected at any given time, but, for
12 simplicity, only 8 computers are shown. As a result of these various
13 unidentified computers connected to the Internet, the computer 12 is very
14 vulnerable to unwanted connections, such as from hackers or transmitters of
15 potentially disabling computer viruses.

16 **[0019]** Although the insecure network shown 10 is preferably
17 connected to the Internet, other types of networks can certainly be used in
18 conjunction with the Internet or even in place of it. For example, the
19 network connection may include other Wide Area Networks (WANs) or
20 even LANs. The present invention can be implemented with any type of
21 network that is considered insecure, and these other implementations should
22 be apparent to one skilled in the art.

23 **[0020]** However, because the network system 10 is contemplated
24 as varying greatly in types, complexity and size, an explanation of the
25 current preferred embodiment of the network topology is given for
26 clarification purposes. Thus, simply as an example, a computer 12 installed

1 with the Microsoft® Windows® operating system having a continuous
2 connection to the Internet (i.e., insecure network) will be used as an
3 example in describing one implementation of the present invention.
4 However, other implementations with different software programs, such as
5 network security programs, network programs or operating systems, are
6 contemplated, and they are considered to be within the scope of the present
7 invention.

8 [0021] Turning to an important aspect of the illustrated
9 embodiment of the present invention, a flow chart of the preferred
10 functionality of the illustrated embodiment of the present invention is
11 shown in FIG. 2, and is indicated generally at 50. The present invention is
12 preferably implemented as an executable software program within the
13 program controlling the connection to the insecure network. However,
14 other implementations, such as firmware or hardware, are contemplated,
15 and it should be understood that these other implementations are considered
16 to be within the scope of the present invention.

17 [0022] At the start of the method (e.g., the execution of the
18 software program implemented with the present invention) (block 52), an
19 address of the network card and the interface connected to the insecure
20 network along with its status are preferably obtained (block 54). The
21 preferred steps of the subroutine for this step (block 54) is shown in FIG. 3.

22 [0023] Turning to FIG. 3, the first step in the Windows®
23 environment is to first initialize the Windows® sockets support or driver
24 (block 56), followed by a step of loading a "INETMIB1.DLL" file or driver
25 (block 58). After this, two addresses for two functions of
26 SNMP_EXTENSIONINIT and SNMP_EXTENSIONQUERY are obtained

1 from the INETMIB1.DLL (block 60). The SNMPEXTENSIONINIT
2 function is then called in order to initialize the INETMIB1.DLL file (block
3 62). After the INETMIB1.DLL is initialized (block 62), the address (e.g.,
4 the object identifier) of a network card (e.g., 1.3.6.1.2.1.2.1.0) is now
5 obtained (block 64). Next, the number of the interface(s) at the address of
6 the network card (e.g., 1.3.6.1.2.1.2.0) is read from the INETMIB1.DLL
7 file (block 66) and stored in memory. The status of the interface is also
8 read at this time at the address or object identifier of the interface (e.g.,
9 1.3.6.1.2.1.2.7.?) (block 68). Note that a question mark (?) has been used to
10 indicate the address of the interface, because the actual address is not
11 known, since the address of the interface is a variable generated at the time
12 when the connection is made. Once all the information is obtained, the last
13 status and the address/object identifier of the interface is then saved into
14 memory (block 70).

15 [0024] Turning back to FIG. 2, after the address of the network
16 card and the interface connected to the insecure network along with its
17 status are obtained (block 54), the next step is to wait for a predefined time
18 period (block 72), which can be implemented according to the computer
19 engineers' desire. Nevertheless, the time out period is preferably less than
20 30 seconds in order to ensure that the computer is constantly checked for
21 deactivation from the insecure network. After waiting for the predefined
22 time out period (block 74), the method 50 will then determine whether there
23 is a network reactivation request (block 74). In the present invention, this
24 command is preferably requested through a user interface by users, but it is
25 also contemplated that other programs in the system may request the
26 network reactivation as well. For example, when a program installed on the

1 computer makes a request to utilize the insecure network, a command to
2 reactivate the network can be generated automatically in the present
3 invention. As a result, even if the user does not directly request the
4 reactivation, it is contemplated that other programs, nevertheless, can
5 trigger the reactivation or deactivation of the insecure network in the
6 present invention. Again, these other various implementations are within
7 the scope of the present invention.

8 [0025] If there is a request to reactivate the network (block 74),
9 the subroutine for reactivating the network (block 76) in the Windows®
10 environment is shown in FIG. 4. Thus, turning for a moment to FIG. 4, the
11 computer can be reactivated by setting the address/object identifier of the
12 interface to "1" for an active status (block 78). Since the reactivation
13 request is preferably generated by the user, it is preferable that a message
14 indicating that the insecure network is active is prompted or displayed on
15 the computer (block 80). From this step, going back to FIG. 2, the process
16 will be repeated from the step to wait for a predefined time (block 72). On
17 the other hand, if there is no network reactivation requested (block 74), the
18 process continues to the next step of determining whether the insecure
19 network indicates an active status (block 82). In other words, the method
20 50 checks to determine whether the insecure network has already been
21 deactivated. If not (block 82), the process will be repeated from the step of
22 waiting for a predefined time (block 72).

23 [0026] Otherwise, if it is determined that the insecure network is
24 currently active (block 82), the process continues to the next step of
25 determining whether there is a network deactivation request in the system
26 (block 84). Similar to the reactivation, any network deactivation is

1 preferably generated from the user interface by users. However, it is also
2 contemplated that the network deactivation request can be generated by
3 other programs in the system. Thus, these various other implementations
4 are contemplated as being within the scope of the present invention. If a
5 network deactivation has been requested (block 84), a network deactivation
6 subroutine (block 86) shown in FIG. 5 will be executed.

7 [0027] Referring to FIG. 5, the first step of the network
8 deactivation subroutine (block 86) executed from FIG. 2 is to set the
9 address/object identifier of the interface to "2" for an inactive status (block
10 88). Since the reactivation request is preferably generated by the user, it is
11 preferable that a message indicating that the insecure network is active is
12 prompted or displayed on the computer (block 90). From this step, going
13 back to FIG. 2, the process will be repeated from the step of waiting for a
14 predefined time (block 72). On the other hand, if there is no network
15 reactivation requested (block 74), the process continues onto the next step
16 of determining the status of the screen saver (block 92). In other words, the
17 screen saver is checked to see if it is activated, and an explanation of the
18 subroutine of this step is shown in FIG. 6.

19 [0028] Turning now to FIG. 6, in order to determine whether the
20 screen saver is active (Block 92), it must be first determined whether the
21 current version of Windows® is running on the computer 12 (block 94),
22 which then separates into three different versions. If the version is not
23 Windows NT®, the "FINDWINDOW" function is executed to find a
24 "WINDOWS-SCRENSAVER" command (Block 96). If the
25 "WINDOWS-SCRENSAVER" command is found (block 98), a
26 determination of the screen saver being active is returned (block 100) back

1 to the process shown in FIG. 2. Otherwise, a determination of the screen
2 saver being not active is returned (block 102) to the process shown in FIG.
3 2.

4 [0029] If it is determined that the current version of Windows® is
5 a NT version that is newer than 4.0 (block 94), a
6 "SYSTEMPARAMETERSINFO" function is executed to find a
7 "GETSCREENSAVERRUNNING" command (block 104). Similarly, if
8 the "GETSCREENSAVER-RUNNING" command is found (block 106), a
9 determination that the screen saver is active is returned (block 108) to the
10 process in FIG. 2. Otherwise, a determination of the screen saver being not
11 active is returned (block 110) to block 122 in FIG. 2.

12 [0030] If it is determined that the current version of Windows® is
13 a NT version 4.0 or older (block 94), there is an attempt to open the desktop
14 of the computer 12 where the screen saver is running on (block 112). If the
15 attempt to open the desktop is successful (block 114), a determination that
16 the screen saver is active is returned (block 116) to block 122 in FIG. 2.
17 Otherwise, it must be determined whether access has been denied by the
18 program (block 118). If, in fact, access has been denied (block 118), a
19 determination of the screen saver being active is returned (block 116). On
20 the other hand, if access has not been denied (block 118), a determination of
21 the screen saver being not active is then returned (block 120).

22 [0031] Turning back to FIG. 2, once it is determined whether the
23 screen saver has been activated (block 92) from FIG. 6, in the case when the
24 screen saver is activated (block 122), the insecure network will be
25 deactivated, which is previously illustrated in FIG. 5. Otherwise, the
26 process continues to the next step of determining whether there is any active

1 network process currently running (block 124), which is explained using
2 FIG. 7.

3 [0032] Turning now to FIG. 7, to determine whether any active
4 network process is currently running on the system in the Windows®
5 environment, the first step is to read an old number of received bytes and
6 transmitted bytes (block 126), which is a number saved from the previous
7 run through the process. If, however, this is the first time the process is has
8 been run, the old number will be preferably zero. Next, the obtained
9 address of the interface/object identifier (e.g., 1.3.6.1.2.1.2.7.?) must be
10 changed to an address/object identifier (e.g., 1.3.6.1.2.1.2.10.?) (block 128)
11 for obtaining or reading the number of bytes received during this process
12 (block 130), which is then saved as a new number (block 132). Similarly,
13 to obtain the number of bytes transmitted, the obtained address of the
14 interface/object identifier (e.g., 1.3.6.1.2.1.2.10.?) is changed to an
15 address/object identifier (e.g., 1.3.6.1.2.1.2.16.?) (block 134) for obtaining
16 or reading the number of bytes transmitted during this process (block 136).
17 The obtained number of bytes transmitted is again saved as a new number
18 (block 138). The old numbers of the received bytes and the transmitted
19 bytes are then compared to the new numbers obtained (block 140). If the
20 old numbers are equal to the new numbers (block 140), a determination that
21 a network process is currently active and running is returned (block 142) to
22 FIG. 2. If, however the old numbers do not equal the new numbers (block
23 140), a determination that a network process is currently active and running
24 is returned (block 144) to FIG. 2.

25 [0033] Finalizing the process, after it is determined whether any
26 active network process is currently running (block 124), the insecure

1 network is deactivated (block 86) if it is determined that an active network
2 process is currently running (block 146). On the other hand, if no active
3 network is currently running in the system (block 146), the process reloops
4 back to wait for a predefined time to restart the process (block 72).

5 [0034] From the foregoing description, it should be understood
6 that an improved method and system for securing a computer connected to
7 an insecure network have been shown and described, which have many
8 desirable attributes and advantages. The present method and system
9 provide a way to completely deactivate the computer from the insecure
10 network when the computer is not utilizing the insecure network. Thus,
11 there is no need to filter the requesting sources, as done in the prior art,
12 because once the computer is deactivated from the insecure network, no
13 data is allowed to be received or transmitted through the insecure network
14 no matter what the requesting source may be. Any communication through
15 the insecure network is completely disabled. As a result, any security leaks
16 or shortcomings in the system would be greatly reduced by the present
17 invention, and network security is improved.

18 [0035] It should be noted that, although a preferred method has
19 been shown with certain order, it would be apparent to one skilled in the art
20 that the order of the steps can be changed, and the steps, themselves, can be
21 slightly altered. In addition, new steps can be added as well. These
22 variations in alternating the preferred method is apparent to one skilled in
23 the art, and the present invention is not limited to the method shown. Thus,
24 it should be understood that other variations of the preferred method shown
25 is contemplated and within the scope of the present invention.

1 [0036] While various embodiments of the present invention have
2 been shown and described, it should be understood that other modifications,
3 substitutions and alternatives are apparent to one of ordinary skill in the art.
4 Such modifications, substitutions and alternatives can be made without
5 departing from the spirit and scope of the invention, which should be
6 determined from the appended claims.

7 [0037] Various features of the invention are set forth in the
8 appended claims.